THE DEVELOPMENT OF ARRAY ARCHITECTURES EMBODYING PARTIAL LOCAL AUTONOMY

T J Fountain

Department of Physics and Astronomy
University College London
Gower Street, London WC1E 6BT

ABSTRACT

SIMD arrays have proved very useful for low-level image processing tasks. The CLIP7 programme is seeking to extend their area of applicability to intermediate- and high-level operations, particularly by increasing the amount of local autonomy for each processor in the array. This paper describes a sequence of degrees of autonomy which are used as the basis of the investigation, and a linear array of partially autonomous processors (CLIP7A) which is the first system to be built in this programme. The system embodies the ability to locally determine the neighbourhood connectivity of each element and the freedom for each element to determine addresses within the local data RAM. Some suggestions are given of algorithms which might use these facilities.

I INTRODUCTION

Flynn's classical taxonomy of computer architectures, embodying the ideas of singular and multiple instruction and data streams, is widely used as the basis of a convenient shorthand method of classifying multiprocessor architectures (Flynn 1966). Two of the principal categories of classification are SIMD, generally used to refer to architectures of the CLIP4 (Fountain and Goetcherian 1980) type, and MIMD, which is used as a catch-all category covering any assembly of processors which act autonomously for some or all of the time.

From the differing degrees of success with which the two types of systems have been employed, and from a consideration of the types of algorithms which can be most effectively applied to each, it is apparent that a wide conceptual gulf exists between the two categories. The CLIP7 programme at University College London is attempting to bridge this gulf by constructing systems which embody increasing degrees of autonomy for each processor, whilst achieving a complete understanding of the operation and application of the system at each stage.

This paper addresses the first stages of this process, namely the analysis of the types of autonomy which processors in an array can embody, together with examples of the operations which each step allows to be more efficiently executed. In the following a progression from pure SIMD to complete MIMD is derived which provides a conceptual framework around which subsequent parts of the CLIP7 programme will develop. The second part of the paper describes CLIP7A, the first system which has been constructed under the Alvey programme to investigate and develop these concepts.

II DEGREES OF LOCAL AUTONOMY

The first, necessary step in the process set out in this paper is the identification, in general terms, of the various levels of autonomy available to each processor in an array of elements which are acting in concert upon some problem. In the following it is assumed that an appropriate global control is provided and that, usually, control of the locally-autonomous aspects can be either global or local as required. No matter how much independence each element is allowed, some degree of local control will remain.

Null

In order to develop a progression of autonomous systems, we should first define a starting-point. Although it ought to be rather easy to propose a processor which embodies no local autonomy, it is, in fact, more difficult than might have been supposed. One example which the author can adduce is that of a single-function systolic array, for example a convolution device (McCanny and McWhirter 1982). Not only is no local autonomy permitted, but the global function cannot be changed.

Activity

Most of the so-called SIMD processor arrays (CLIP4, DAP (Reddaway 1973), MPP (Batcher 1980), etc.) do in fact embody very simple local autonomy, namely the ability to locally determine whether or not each processor in the array is active. This can be easily achieved either by selective loading of results or by logically including a masking image in the computed function, and leads to substantial improvements in performance for some algorithms. A particular example of this occurs when computation of a result is required only within specified areas of an image, in which case the use of local autonomy offers a twofold improvement in performance.

Data

Processors which embody this type of autonomy allow the source and destination of the data used by a globally-determined process to be locally selected. This can be achieved either by calculating a local address (or offset) for the local memory, or by local control of the neighbourhood connectivity functions. In either case the processor function is under global control. Algorithms whose performance might be improved by this facility include the mapping of one structure onto another (e.g. a graph onto a mesh), or those where selections from different areas of the data space are required.

Function

The elemental processors in an array can be regarded as consisting of two sections, one part concerned with movement of data around the processor, the other part executing some function upon that data. The function generator may be singular (e.g. an 8-bit ALU) or multiple (e.g. ALU, barrel-shifter and multiplier). In either case the next stage of local autonomy allows each processor to select which function is to be performed, data movement being under global control. This can be achieved either by storing a previous result and using that to control the function, or by local addressing of a look-up table of functions. Specific examples of how this type of local control might be used are confined at present to improvements in the efficiency of arithmetic operations such as the normalisation of arrays of floating-point numbers.

Algorithm

In this mode of local autonomy, each processor in the array is provided with a section of program control store. However, the sequencing of these stores is under global control, thereby ensuring synchronous operation of the array. The size of the program store may be small, in which case only a single algorithm is stored at each processor. The processors can have different algorithms and synchronous operation is ensured by appropriate padding, all algorithms comprising the same number of micro-instructions. The same is true if the program store is large, but in this case a library of algorithms can be stored at each processor, selection being by local addressing. One type of system which might benefit from this degree of local autonomy is the programmable systolic array. The algorithms implemented on such arrays frequently require synchronous formatting and reformatting of data at the edges which proceeds simultaneously with the desired computation in the body of the array. One example of a system used in this fashion is the PSC array (Fisher et al. 1983).

Sequencing

Each processor in this type of array not only has local selection of algorithm, but is also able to sequence its own program memory. This in turn implies that synchronism over the array is no longer automatic, but must be ensured by appropriate handshaking operations. At this level, only data needs to be passed between processors. It is likely that arrays using this degree of autonomy would be particularly efficient for graphics operations such as ray tracing, where the time taken to compute results for various areas of data varies randomly for each processor.

Partitioning

At this highest level of local autonomy, processors not only autonomously manipulate data which may be passed between them, but can alter the placing of program segments within the array, presumably on the basis of load equalisation or as a result of sequencing a complex program of the expert system sort.

III THE CLIP7A SYSTEM

The long-term intention of the CLIP7 programme is to investigate these and other ideas by constructing a series of prototype systems from CLIP7 chips (Fountain 1987). The chip, whose data paths are shown in Figure 1, contains a single 16-bit processing element with flexible arrangements for neighbourhood connectivity which allow it to be built into a variety of array configurations. Conditional operations are achieved by the use of a 16-bit condition register, which can be loaded either with the present status of the processor or with local data. This register may affect the operation of the ALU, enable the loading of various registers or be used as the addressing source of other registers.

The chip itself is designed in a conservative 5 μm, 5 MHz CMOS technology which results in a time of 800 nsec for the addition of two 8-bit numbers stored in external memory. The die, comprising about 7000 transistors, has an area of around 20 mm^2 and is available in either 64-pin or 68-pin packages.

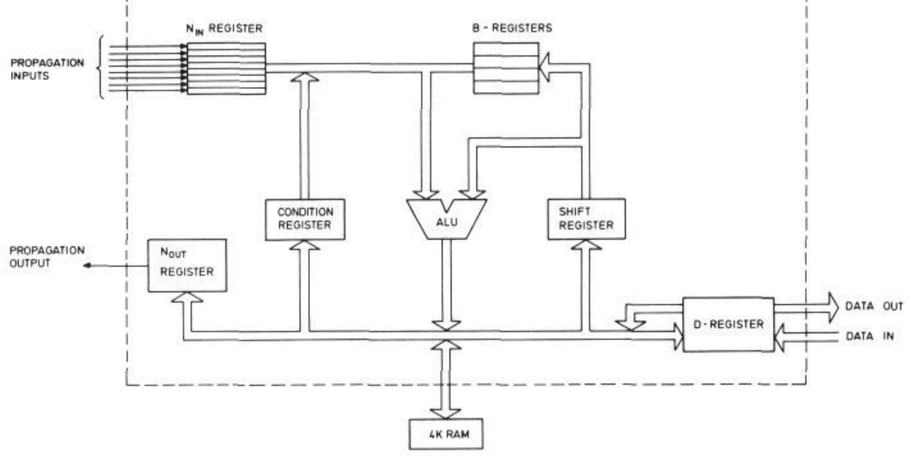

Figure 1. The CLIP7 chip data paths.

A. Overall Configuration

The first system to be constructed from CLIP7 chips is a linear array of processing elements called CLIP7A. The main reasons for choosing this structure were as follows:

(a) A linear array can easily be used to emulate a two-dimensional planar array when low-level image processing operations are required.

(b) A linear array can be used as a vector processor when manipulation of (for example) symbol strings resulting from low-level processing is needed.

(c) The relatively simple structure of the linear array should facilitate its use in the simulation of other more complex structures.

(d) A linear array is relatively cheap and easy to construct.

The major elements of the system are a Sun-3 workstation, which acts as the system host, a controller which is a microcode sequencer, a CRS4000 TV data interface and frame store system, and the 256-element linear array. The array size was chosen to permit easy processing of 256 by 256 element images, which are of suitable size for both algorithm generation and feasibility studies to be carried out. A microcode sequencer is used as the controller principally because of the flexibility which it allows. For a prototype system such as CLIP7A, whose principal use will be research into, and development of, algorithms, a complete instruction set cannot be predefined, so a degree of flexibility in the controller is required. The unit developed for CLIP7A (Matthews 1986) incorporates a 16,000 word microcode RAM (word width 160 bits), a microcode sequencer and a number of special-purpose registers which deal with the interchange of data between the array and the host.

The Sun-3 workstation was chosen as host for the system because it offered a combination of Berkeley 4.2 UNIX, VME bus hardware and a 16 MHz MC68020 processor. This combination meant that it would be able to support the interface speed required by the CLIP7A array/controller.

B. The Processing Element

The CLIP7 chip itself embodies local control of activity, neighbourhood connectivity and function, and it seemed worthwhile to incorporate, in the CLIP7A PEs, local generation of the data memory address. This was achieved as shown in Figure 2 by using two CLIP7 chips operating in tandem; one mainly concerned with data manipulation and the other principally dealing with address generation.

C. System Status

The CLIP7A system construction is now complete and substantial sections have been debugged. It is estimated that, by the date of the AVC87 meeting, the system, including intermediate-level software, will have been in use for some time. Initial investigations will concern the degrees of autonomy which have been built into the system but, subsequently, simulations of the more complex degrees of local autonomy described above will be implemented.

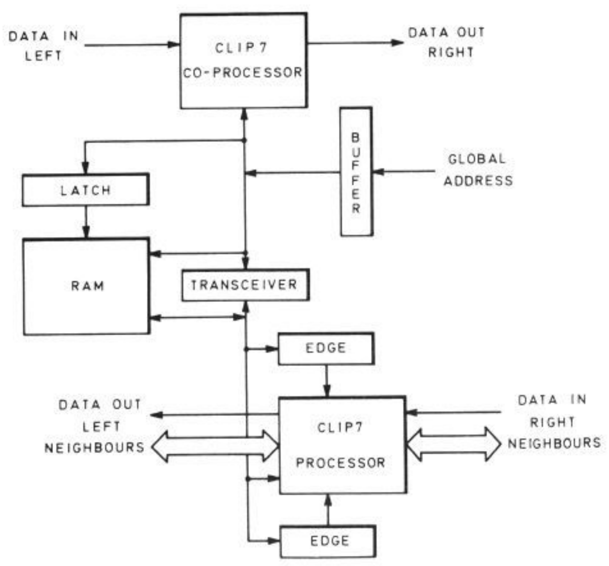

Figure 2. The CLIP7A processing element.

IV LOCALLY-AUTONOMOUS ALGORITHMS

The hardware system embodies activity, data and partial function control. It can therefore be directly used either in pure SIMD mode or with any (or all) of these types of local autonomy. At each (increasing) level of autonomy, two classes of algorithms will be assessed. The first class comprises those functions whose efficiency should be improved by utilising the current level, i.e. operations where an acceptable algorithm exists at the previous level, but the additional autonomy offers a better method. Two examples of such operations relevant to the level of autonomy built in to CLIP7A are as follows. The first concerns segmentation of objects from background, a frequently-required (and often difficult) preliminary operation. For a variety of reasons, commonly-used segmentation algorithms yield incomplete object boundaries. A number of purely SIMD techniques are available for completing these boundaries by joining nearest pairs of points, but the resulting additional boundary segments comprise straight lines which often do not conform well to the ideal boundary. If line ends could be extended with the same direction and curvature as the existing line segment, then a better approximation to the true boundary could be achieved. The ability of each processor to control its own neighbourhood connectivity would permit this to occur for all missing segments simultaneously.

The second example concerns the use of the local addressing facility. It is often the case that, when processing a large image containing a number of small, complex, objects, a simple low-cost algorithm can be used to determine the position of the objects, whereupon attention can be focussed upon each object for processing with more complex, costly algorithms. For a linear array such as CLIP7A scanning two-dimensional images, the local addressing facility can be used to direct different groups of processors to different areas of the data field simultaneously, thereby improving efficiency.

Such algorithms should give a direct measure of the benefits offered at each level, simply by comparing performance and perhaps ease of programming. The second class consists of those operations which would not have been contemplated at the previous level, but which are facilitated by the additional autonomy. Functions of this sort which are relevant to the first stage of use of CLIP7A are the mapping and searching of graph structures, perhaps in the context of expert systems, which would utilise the local control of neighbourhood connectivity, and the locally-directed search of vector database structures. These structures can often result from the low-level processing of sets of medical images (e.g. electrophoresis gels) and the ability to locally control data addresses should help in the search and analysis of the data. This class gives a less direct measure of comparative benefits, but does indicate the increasing range of problem-solving capabilities offered by increasing autonomy.

As with any other exercise in simulation, both implementation and analysis of results when this technique is employed will be much more difficult. In this phase of the investigation, the only relevant benefits of employing CLIP7A will be its flexible control structure and its comparatively high power. It should be possible, however, to obtain a sufficient idea of the benefits of further increases in local autonomy to guide the design of subsequent systems, if indeed these are deemed worthwhile.

V SUMMARY

One of the main aims of the CLIP7 programme is to develop an understanding of the role of local autonomy in large arrays of processors. The method chosen to approach this task is to proceed stepwise through a sequence of increasing degrees of processor autonomy, described above. It is envisaged that results from the earlier steps will modify the later stages of the process. The vehicle for this investigation is the CLIP7 chip and, initially, the CLIP7A system outlined here.

REFERENCES

Batcher, K. E. "Design of a Massively Parallel Processor." *IEEE Trans.* C-29 (1980) 836-840.

Fisher, A. L., H. T. Kung, L. M. Monier, H. Walker and Y. Dohi "Design of the PSC: A Programmable Systolic Chip." In *Proc. 3rd CALTECH Conf. on VLSI*, 1983, pp. 287-302.

Flynn, M. J. "Very High Speed Computing Systems." *Proc. IEEE* 54 (1966) 1901-1909.

Fountain, T. J. *Processor Arrays: Architectures and Applications*. London: Academic Press, 1987.

Fountain, T. J. and V. Goetcherian "CLIP4 Parallel Processing System." *IEE Proc.* 127E (1980) 219-224.

Matthews, K. N. "The CLIP7 Image Analyser - A Multi-Bit Processor Array." PhD Thesis, University of London, 1986.

McCanny, J. V. and J. G. McWhirter "On the Implementation of Signal Processing Functions Using One-Bit Systolic Arrays." *Electron. Lett.* 18 (1982) 241-243.

Reddaway, S. F. "DAP - A Distributed Array Processor." In *Proc. 1st Annual Symp. on Computer Architecture.* Florida, USA, 1973, pp. 61-65.